РЕШЕНИЕ ЗАДАЧИ РАЗБИЕНИЯ МНОЖЕСТВА МЕТОДОМ РЕЛАКСАЦИИ К СЕТЕВОЙ ЗАДАЧЕ СПЕЦИАЛЬНОГО ВИДА

УДК 330.46
ВАК 05.13.00
ГРНТИ 28.29.00

Геннадий Александрович Беркетов,
к.т.н., профессор,
Российский экономический университет им. Г.В. Плеханова
Тел.: (495) 442-61-11
Эл. почта: GABerketov@mesi.ru

Андрей Александрович Микрюков,
к.т.н., доцент
Российский экономический университет им. Г.В. Плеханова
Тел.: (495) 442-61-11
Эл. почта: AMikrukov@mesi.ru

Анатолий Иванович Полоус,
к.т.н., доцент
Российский экономический университет им. Г.В. Плеханова
Тел.: (495) 442-61-11
Эл. почта: APolous@mesi.ru

В статье рассматривается оригинальный алгоритм решения задачи о разбиении множества, который имеет многочисленные приложения при анализе и синтезе организационно-технических и экономических систем и процессов. Эффективность предлагаемого алгоритма позволяет решать с его помощью характерные для практики задачи большой размерности. Инструментарий решения подобных задач используется в системах поддержки принятий решений для распределения нагрузки в многомашинных и многопроцессорных системах.

Ключевые слова: *задача разбиения множества, системы поддержки принятий решений, метод ветвей и границ, максимальный поток в сети.*

THE SET PARTITION PROBLEM SOLUTION USING RELAXATION METHOD TO THE SPECIALITY NETWORK PROBLEM

Gennadij A. Berketov,
PhD in Technical science, Professor
Plekhanov Russian University
of Economics
Tel.: (495) 442-61-11
E-mail: GABerketov@mesi.ru

Andrej A. Mikrukov,
PhD in Technical science, Associate professor
Plekhanov Russian University
of Economics
Tel.: (495) 442-61-11
E-mail: AMikrukov@mesi.ru

Anatolij I. Polous,
PhD in Technical science, Associate professor
Plekhanov Russian University
of Economics
Tel.: (495) 442-61-11
E-mail: APolous@mesi.ru

The article considers the original algorithm for solving the partition set problem, which has numerous applications in the analysis and synthesis of organizational, technical and economic systems and processes. Efficiency the proposed algorithm allows to solve with its help specific practices for large-scale problems. Instruments for solving such problems included in the decision support system in multicomputer and multiprocessor systems.

Keywords: *set partition problem, branch and bound method, maximum flow in network, decision support system.*

1. Введение

Данная задача имеет многочисленные приложения. К ней, например, сводятся многие экстремальные задачи комбинаторного типа, например, задача оптимизации распределения вычислительных ресурсов в многомашинных и многопроцессорных вычислительных комплексах. Обзор приложений и методов решения ЗРМ приведен в [1,2].

Задача разбиения множества (ЗРМ) может быть записана следующим образом:

$$\sum_{j\in N} C_j x_j \to \min, \tag{1}$$

$$\sum_{j\in N} a_{ij} x_j = 1, i \in M = \{1,...,m\}, \tag{2}$$

$$x_j = 0 \vee 1, j \in N = \{1,\ldots,n\}, \tag{3}$$

где C_j – затраты некоторого ресурса для j-го элемента, $a_{ij} = 0 \vee 1$, i $\in M$, j $\in N$.

В настоящей статье предлагается новый эффективный метод решения ЗРМ, основанный на её сведении к сетевой задаче специального вида.

2. Построение моделирующей сети и сведение решаемой задачи к задаче о максимальном потоке.

Пусть $M_j = \{i \in M | a_{ij} \neq 0\}$. Для построения моделирующей сети возьмем m вершин по одной для каждого ограничения вида (2) и добавим к ним n вершин по одной для каждой переменной x_j. Вершины первого вида будем обозначать с помощью индекса i, а вершины второго вида будем обозначать с помощью индекса j.

Для каждого $i \in M_j$ построим дугу (j, i) с единичной пропускной способностью и нулевой ценой. Введем две дополнительные вершины: S – источник и t – сеток. Каждую вершину $j \in N$ соединим с источником S дугой (S, j), которой припишем пропускную способность $\alpha_{Sj} = |M_j|$ в цену, равную коэффициенту c_j, деленному на пропускную способность $C_{Sj} = C_j/\alpha_{Sj}$. Каждую вершину $i \in M$ соединим со стоком t дугой (i, t) с единичной пропускной способностью и нулевой ценой. Построенную таким образом сеть обозначим через $G(Q, U)$, где $Q = \{S, t\} \cup N \cup M$ – множество вершин, а U – множество дуг. Через α_{ij} и C_{ij} будем обозначать соответственно пропускную способность и стоимость дуги $(i, j) \in U$.

Рассмотрим следующую задачу об оптимальном потоке:

$$\sum_{(i,j)\in U} C_{ij} f_{ij} \to \min, \tag{4}$$

$$\sum_{j\in\Gamma(i)} f_{ij} - \sum_{j\in\Gamma^{-1}(i)} f_{ji} = \alpha_i, i \in Q, \tag{5}$$

$$0 \le f_{ij} \le \alpha_{ij}, (i, j) \in U, \tag{3}$$

где f_{ij} – поток по дуге (i, j), $\Gamma(i) = \{j \in Q / (i, j) \in U\}$, α_i – интенсивность потока из вершины $i \in Q$: $\alpha_S = -\alpha_t \neq 0$; $\alpha_i = 0$, $\forall_I \neq S, t$.

Дополнительно положим: поток по дугам (S, j) равняется или нулю, или пропускной способности этих дуг, т.е.

$$f_{Sj} = 0 \vee \alpha_{Sj}, j \in N \qquad (7)$$

Кроме того, потребуем насыщения дуг, идущих в сток t.

Это означает выполнение равенства

$$\alpha_S = -\alpha_t = |M|. \qquad (8)$$

Легко видеть, что сетевая задача (4) – (8) эквивалентна исходной задаче, так как приписывание дуге (S, j), $j \in N$ потока, равного нулю или её пропускной способности приводит к такому же решению, как приписывание переменной x_j значения 0 или 1 соответственно.

3. Алгоритмы решения основной и вспомогательной задачи

Для решения задачи (1) – (3) можно применить алгоритм, построенный на основе метода ветвей и границ [1,5]. Опишем способ вычисления нижней оценки, используемой в алгоритме.

Пусть f^* – оптимальный поток для задачи (4) – (8) и $C(f^*)$ – и его стоимость. Оптимальный поток по сети G, полученный без учета ограничения (7), обозначим через f^*, а его стоимость – через $C(f^*)$. Очевидно, что $C(f) \geq C(f^*)$.

Оценка $C(f^*)$ для стоимости потока f^* может быть вычислена с помощью известного алгоритма поиска максимального потока минимальной стоимости [3]. Специальная структура сети G позволяет упростить общий алгоритм и значительно сократить необходимый объем вычислений.

Поток по сети G полностью определяется потоком по дугам (S, j), $j \in N$, и (i, t), $i \in M$. Обозначим через F текущее множество вершин $j \in N$, для которых поток по дугам (S, j) фиксирован; положим $H = N / F$. Обозначим, далее, через R текущее множество вершин $i \in M$, для которых дуги (i, t) не насыщены, т.е., $f_{it} = 0$. В начале работы алгоритма $H = N$, $R = M$. Упорядочим элементы множества H по возрастанию $C(S, j)$ стоимости дуг (S, t). Для этой цели можно использовать один из алгоритмов быстрой сортировки, например, алгоритм Шели [4]. Естественным представлением множества H является стек. Операцию выделения первого элемента множества H (вершины стока) обозначим через $V(H)$. Для упрощения описания алгоритма введем символ «←», который будем использовать как знак присваивания значения переменной.

Ниже приведен алгоритм решения оценочной задачи (алгоритм 1).

Алгоритм 1

1. Если $H = \varnothing$ – перейти к п.4, иначе $k \leftarrow V(H)$.

2. $M_k^* \leftarrow M_k \cap R$.

Если $M_k^* = \varnothing$, то

$f_{SK} \leftarrow 0$, $H \leftarrow H \backslash M_k^*$ и перейти к п.1.

3. $f_{SK} \leftarrow |M_k^*|$, $H \leftarrow H \backslash \{k\}$,

$R \leftarrow R \backslash M_k^*$, $C \leftarrow C + C_{SK} \times f_{SK}$.

Перейти к п.1.

4. Конец алгоритма. $C(f^*) = C$.

Для решения задачи (1) – (3) методом ветвей и границ необходимо определить: а) схему ветвления дерева вариантов; б) способ вычисления нижней оценки вариантов; в) метод движения к оптимуму.

Рассмотрим отдельно каждый из этих компонентов алгоритма.

1. Схема ветвления. Введем дерево вариантов T, которое организовано следующим образом. Каждая вершина имеет $2(n - k)$ соседних снизу вершин, где k – номер яруса дерева вариантов. Обозначим эти вершины символами $+x_{k+1} - x_{k+1}, \ldots, + x_n, -x_n$ (n – число переменных в задаче). Дерево T имеет $n + 1$ ярусов (уровней). Процесс поиска решения интерпретируется как процесс движения по дереву T. Условимся, что при попадании в вершину $+x_j$, переменная $x_j = 1$, а при попадании в вершину $-x_j$, переменной x_j присваивается значение 0. Пусть S – путь по дереву, которому соответствует некоторая цепочка символов $+x_j$ или $-x_j$. Такие цепочки символов будем называть частичными решениями. Частичные решения, соответствующие конечным вершинам дерева, называются допустимыми решениями. Каждому допустимому решению соответствует значение целевой функции $C(S)$. S^* – оптимальное решение, если

$C(S^*) \leq C(S)$ для всех допустимых S.

Вычисление оценки. На произвольном шаге алгоритма имеем фиксированные (со значением 0 либо 1) переменные x_j, вошедшие в S, и некоторую ЗРМ вида:

$$\sum_{j \in N_S} C_j x_j \rightarrow \min, \qquad (9)$$

$$\sum_{j \in N_S} \alpha_{ij} x_j = 1, \forall i \in M_S^*, \qquad (10)$$

$$x_j = 0 \vee 1, \forall j \in N_S, \qquad (11)$$

где $N_S = N \setminus \overline{N_S}$, $\overline{N_S} = \{j \in N \mid x_j \in S$ либо $\bar{x}_j \in S\}$, $M_S^* = \{i \in M \mid \sum_{j \in N_S} \alpha_{ij} \geq 1\}$.

Для вычисления оценки условие (11) заменяется условием

$$0 \leq x_j \leq 1, \forall j \in N_S. \qquad (12)$$

Задача (9,10,12) решается с помощью алгоритма 1. Минимальное значение $C(x_s)$, $x_s = \{x_j | j \in N_s\}$ обозначим через C_s.

3. Поиск оптимального решения. Алгоритм не требует запоминания дерева вариантов: вся информация, необходимая для вычислений, запоминается в виде текущего частичного решения. В процесс поиска дерево Т обходится в последовательно устанавливаемом порядке. При этом, могут встретиться два случая: когда рассматриваемый путь продолжается и когда его продолжение следует прекратить и выполнить «шаг назад», т.е. вернуться к ранее пройденной вершине. Такое возвращение выполняется в следующих случаях: 1) становится известным, что данный путь не может быть продолжен до оптимального пути задачи, т.е. оценка снизу значения целевой функции на всех последующих продолжениях пути превышает значение рекорда; 2) данный путь является допустимым решением, в этом случае нужно сравнить полученное решение с рекордным и, если оно лучше, запомнить его в качестве нового рекордного решения.

Для формального описания алгоритма введем следующие обозначения:

H – текущее множество (список) свободных переменных, упорядоченное в порядке возрастания C_j, в начале все $x_j \in H$;

R – верхняя граница (рекорд) для минимума задачи;

$\underline{C}$ – нижняя граница (оценка) значения целевой функции на всех допустимых продолжениях пути S;

S_R – рекордное решение.

На первых шагах алгоритма считаем, что $R = +\infty$, в дальнейшем полагаем R равным минимальному значению целевой функции на множестве найденных допустимых решений. В начале алгоритма $S = \varnothing$.

Ниже приведен алгоритм решения задачи (1–3).

Алгоритм 2

1. Выбирается переменная $x_j \in H$ с наименьшим номером j, т.е. с наименьшим значением коэффициента C_j. Переменной x_j присваивается значение 1, т.е. x_j вычеркивается из H и приписывается со знаком (+) к правому концу S. Если $H = \varnothing$ (S – допустимое решение), то $S_R \leftarrow S$ и перейти к п.4.

2. Исключаются все пути дерева, которые не ведут к допустимому решению. Для этого определяется множество зависимых переменных

$$x_j = \left\{ x_k \in H \middle| \alpha_{ik} + \alpha_{ip} \geq 1, \forall i \in M, \forall p \in \overline{N_s} \right\}$$

Всем $x_k \in X_j$ присваивается значение 0, т.е. переменные x_k вычеркиваются из H и приписываются справа к S со знаком (–).

3. Реализуется основная идея метода ветвей и границ. Вычисляется оценка $\underline{C}(S)$ наименьшего значения C, которое можно получиться, двигаясь «вниз» по дереву из вершины x (x – последняя вершина в частичном решении S).

Для этого решается оценочная задача (9) – (12) и полагается

$$\underline{C}(S) = C(S) + C_S$$

Если $\underline{C}(S) < R$ – перейти к п.1. Если $\underline{C}(S) \geq R$ или оценочная задача не имеет решения, то осуществляется переход к следующему пункту.

4. Осуществляется возврат к еще не исследованным ветвям дерева T. Список S просматривается в обратном направлении и находится первый символ x_t со знаком (+), заменяется его значение на противоположное. Все символы правее его удаляются из S и переносятся в H, после чего осуществляется переход к п.1. Если таких символов в S нет, алгоритм заканчивает свою работу. Если $R < \infty$, то S_R – оптимальное решение задачи (1) – (3). В противном случае задача не имеет решения.

5. Алгоритм 2 либо находит (за конечное число шагов) решение, либо сообщает, что решений нет.

Заключение

В статье рассмотрен подход к решению задачи разбиения множества, основанный на ее сведении к сетевой задаче специального вида. Предложен алгоритм решения задачи, обладающий более высоким быстродействием по сравнению с известными, что позволяет решать с его помощью характерные для практики задачи большой размерности. Для построения алгоритма использована модифицированная схема метода ветвей и границ, особенностью которой является вычисление нижней оценки частичных решений на основе нахождения максимального потока в сети специального вида. Результаты вычислительного эксперимента подтвердили высокое быстродействие предложенного алгоритма.

Литература

1. Бурков В.Н., Ловецкий С.Е. Методы решения экстремальных комбинаторных задач (обзор). – «Извест. АН СССР.Техническая кибернетика», №4, 1969.

2. Трубин В.А. О методе решения целочисленного линейного программирования специального вида. – «Докл. АН СССР», 189, №5, 1969.

3. Кристофидес Н.К. Теория графов. Алгоритмический подход. «Мир», 1978.

4. Э. Рейнгольд, Ю. Нивергельт, Н. Део. Комбинаторные алгоритмы. «Мир», 1980.

5. Беркетов Г.А., Микрюков А.А., Федосеев С.В. Оптимизация технологических процессов обработки информации в АСУ// Сб. трудов Международной научно-практической конференции «Инновации в условиях развития информационно-коммуникационных технологий «Инфо-2008». – Сочи, 2008. – С. 197–200.

References

1. Burkov V.N.. Lovetskiy S.E. Metody resheniya ekstremalnykh kombinatornykh zadach (obzor). – «Izvest. AN SSSR.Tekhnicheskaya kibernetika». №4. 1969

2. Trubin V.A. O metode resheniya tselochislennogo lineynogo programmirovaniya spetsialnogo vida. – «Dokl. AN SSSR». 189. №5. 1969.

3. Kristofides N.K. Teoriya grafov. Algoritmicheskiy podkhod. «Mir». 1978.

4. E. Reyngold. Yu. Nivergelt. N. Deo. Kombinatornyye algoritmy. «Mir». 1980.

5. Berketov G.A., Mikryukov A.A., Fedoseyev S.V. Optimizatsiya tekhnologicheskikh protsessov obrabotki informatsii v ASU// Sb. trudov Mezhdunarodnoy nauchno-prakticheskoy konferentsii «Innovatsii v usloviyakh razvitiya informatsionno-kommunikatsionnykh tekhnologiy «Info-2008». – Sochi. 2008. – S. 197–200.